ADSP-CM403 HAE在太阳能应用中的谐波分析

作者：Martin Murnane

ADI太阳能光伏发电系统

martin.murnane@analog.com

简介

太阳能光伏逆变器转换来自太阳能面板的电能并高效地将其部署到公用电网中。早期太阳能PV逆变器只是将电能转储到公用电网的模块。但是，新设计要求太阳能光伏逆变器对电网的稳定性作出贡献。

本文将回顾最新的ADI技术如何以HAE(谐波分析引擎)的方式改善智能电网的集成度，并监控电网上的电源质量，从而极大地增强电网稳定。

智能电网

什么是智能电网？IMS Research将智能电网定义为“一种自身能够高效匹配和管理发电和用电并可最大程度地利用各种可用资源的公用供电基础设施”。若要将新一代太阳能光伏逆变器接入智能电网，则逆变器需要越来越高的智能程度才能实现。这本身就是一个难题，主要是因为当电力需求在别处时，此处却连接了过多的电网，从而发生不平衡。基于这个原因，如前文所述，太阳能光伏逆变器需要具备更高的智能程度，并且这种智能应侧重于电网集成，其中系统需协助稳定电网，而非作为电网的一个简单电源使用。

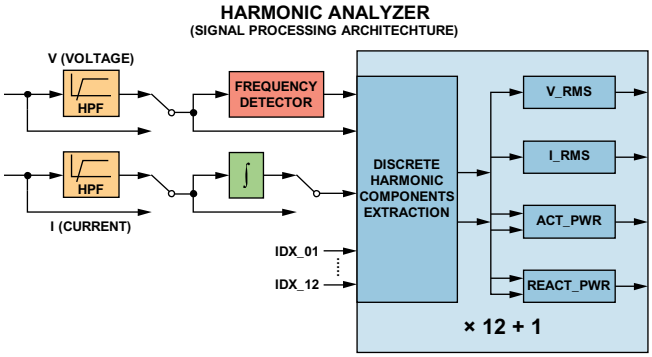

图1. ADSP-CM403 HAE框图(ADI公司)

这要求更好地对注入电网的电能进行测量、控制和质量分析。当然，这会促成新指令的发布以及更高的技术要求，进而直接导致新技术的产生。

ADSP-CM403XY HAE外设模块

HAE模块本质上是一个数字PLL，其简化原理图如下图所示。HAE连续接收V和I数据，并且数个周期后将锁定至输入波形的基波。HAE模块的输入范围为45 Hz至66 Hz。最多可分析40个谐波，每次12个。对于每个谐波，PLL会试图锁定至所需的信号频率。

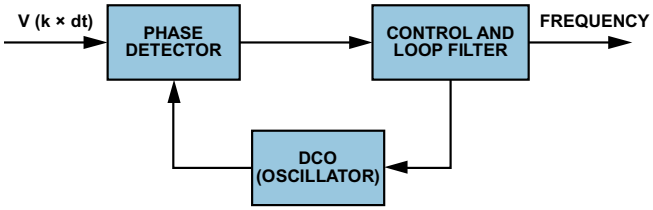

图2. HAE简化数字PLL

谐波引擎硬件模块与谐波分析仪共同处理结果。由于谐波引擎产生的结果为最终格式，这些结果数据保存在结果存储器中。HAE引擎在无衰减的2.8 kHz通带内计算谐波信息(相当于3.3 kHz的-3 dB带宽)，用于45 Hz至66 Hz范围内的线路频率。

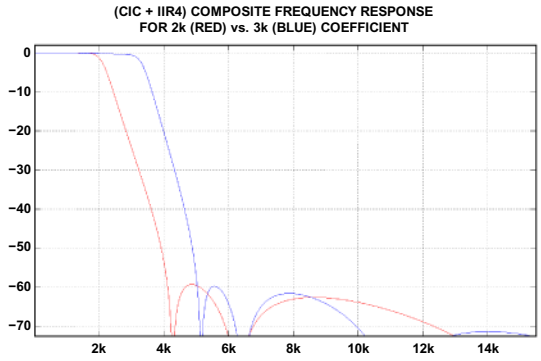

图3. HAE通带频率

同时可使用相电流和来分析零线电流。在新采样周期的最初时刻，谐波引擎在含有数据RAM内的预定义位置循环，该数据RAM含有分析仪处理结果。若有需要，内容可进一步处理。

电压和电流数据可来自Sinc模块或ADC(两者均存储在SRAM中)，并输入至HAE模块，速率为8 kHz。该速率下可产生一个中断，提示太阳能光伏逆变器输入可用数据。进行数据分析并执行下列计算时，HAE模块将产生另一次中断，提示太阳能光伏系统准备显示谐波分析数据。ADSP-CM403还可将HAE至DMA的全部结果数据直接传输至SRAM，之后系统代码便可显示结果。这会导致整个HAE系统的少许代码开销。

ADSP-CM403XY HAE结果

图4中的HAE结果清楚表明观察电压均方根数据时，系统中存在哪些谐波。图中50 Hz基波清晰可见，但250 Hz和350 Hz处的较低谐波(如谐波5和7)亦可在本示例结果中看到。

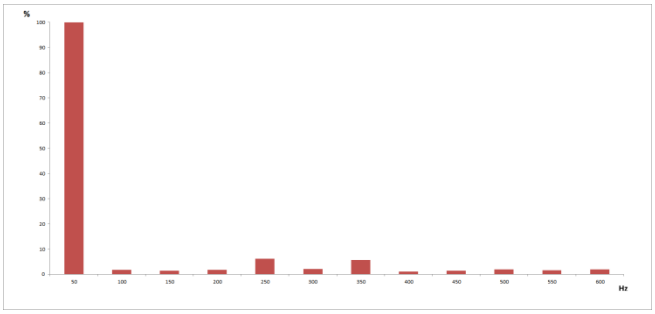

图4. HAE的V rms示例结果(谐波1-12)

这些计算中采用的特定等式如下所示；下列等式同时适用于基波和谐波计算。

资源

分享本文 facebook twitter

表1. HAE数学计算

Harmonic Engine Outputs and Registers where Values are Stored

Quantity	Definition	HAE Registers
RMS of the Fundamental Component	V_1, I_1	F_VRMS, F_IRMS
RMS of a Harmonic Component	V_n, I_n, n = 2, 3,…, 12	Hnn_VRMS, Hnn_XIRMS
Active Power of the Fundamental Component	$P_1 = V_1 I_1 \cos(\varphi_1 - \gamma_1)$	F_ACT
Active Power of a Harmonic Component	$P_n = V_n I_n \cos(\varphi_n - \gamma_n)$, n = 2, 3,…, 12	Fnn_ACT
Reactive Power of the Fundamental Component	$Q_1 = V_1 I_1 \sin(\varphi_1 - \gamma_1)$	F_REACT
Reactive Power of a Harmonic Component	$Q_n = V_n I_n \sin(\varphi_1 - \gamma_1)$, n = 2, 3,…, 12	Hnn_REACT
Apparent Power of the Fundamental Component	$S_1 = V_1 I_1$	F_APP
Apparent Power of a Harmonic Component	$S_n = V_n I_n$, n = 2, 3, …, 12	Hnn_APP
Power Factor of the Fundamental Component	$pf_1 = \mathrm{sgn}(Q_1) \times \frac{P_1}{S_1}$	F_PF
Power Factor of a Harmonic Component	$pf_n = \mathrm{sgn}(Q_n) \times \frac{P_n}{S_n}$, n = 2, 3,…, 12	Hnn_PF
Harmonic Distortion of a Harmonic Component	$HD_{V_n} = \frac{V_n}{V_1}, HD_{I_n} = \frac{I_n}{I_1}$, n = 2, 3,…, 12	Hnn_VHDN, Hnn_IHDN

编程示例

```
INT HAE_CONFIG(VOID)
{  INT I;


 HAE_INPUT_DATA(VOUTPUT, SINC_VEXT_DATA);
 HAE_INPUT_DATA(IOUTPUT, SINC_IMEAS_DATA);


 RESULT = ADI_HAE_OPEN(DEVNUM, DEVMEMORY, MEMORY_SIZE, &DEV);
 RESULT = ADI_HAE_REGISTERCALLBACK(DEV, HAECALLBACK, 0);
 RESULT = ADI_HAE_SELECTLINEFREQ(DEV, ADI_HAE_LINE_FREQ_50);
 RESULT = ADI_HAE_CONFIGRESULTS(DEV,  ADI_HAE_RESULT_MODE_IMMEDIATE,  ADI_HAE_SETTLE_TIME_512,                ADI_HAE_UPDATE_RATE_128000);
 RESULT = ADI_HAE_SETVOLTAGELEVEL (DEV, 1.0);
 RESULT = ADI_HAE_ENABLEINPUTPROCESSING(DEV, FALSE, FALSE); /* FILTER ENABLED */
 /* ENABLE ALL HARMONICS (IN ORDER) */
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_1, 1);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_2, 2);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_3, 3);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_4, 4);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_5, 5);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_6, 6);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_7, 7);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_8, 8);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_9, 9);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_10, 10);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_11, 11);
 RESULT = ADI_HAE_HARMONICINDEX (DEV, ADI_HAE_HARMONIC_INDEX_12, 12);


 RESULT = ADI_HAE_SUBMITTXBUFFER(DEV, &TXBUFFER1[0], SIZEOF(TXBUFFER1));
 RESULT = ADI_HAE_SUBMITTXBUFFER(DEV, &TXBUFFER2[0], SIZEOF(TXBUFFER2));
 RESULT = ADI_HAE_ENABLEINTERRUPT(DEV, ADI_HAE_INT_RX, TRUE);
 RESULT = ADI_HAE_ENABLEINTERRUPT(DEV, ADI_HAE_INT_TX, TRUE);
 RESULT = ADI_HAE_CONFIGSAMPLEDIVIDER(DEV, 100000000);
 RESULT = ADI_HAE_RUN(DEV, TRUE);
//  RESULT = ADI_HAE_CLOSE(DEV);
}

/* EVENTS */
VOID HAECALLBACK(VOID* PHANDLE, UINT32_T EVENT, VOID* PARG)                  /* ISR ROUTINE TO LOAD / UNLOAD DATA FROM HAE
{
   UINT32_T N;
   ADI_HAE_EVENT EEVENT = (ADI_HAE_EVENT)EVENT;              /* RESULTS RECEIVED FROM HAE  128MS */
   IF (EEVENT == ADI_HAE_EVENT_RESULTS_READY)
   {                                                                                                                  /* GET RESULTS */
      PRESULTS = (ADI_HAE_RESULT_STRUCT*)PARG;                                  /* POINTER TO TXBUFFER1 OR TXBUFFER2 */
 /* DO SOMETHING WITH THE RESULTS */
      FOR (N=0; N<NUM_CHANNELS; N++)
      {
        IRMS[N] = PRESULTS[N].IRMS;

        VRMS[N] = PRESULTS[N].VRMS;
        ACTIVEPWR[N] = PRESULTS[N].ACTIVEPWR;
      }
   }                                                                                                                  /* TRANSMIT INPUT SAMPLES TO HAE – 8KHZ */
   IF (EEVENT == ADI_HAE_EVENT_INPUT_SAMPLE)
   {                                                                                                                  /*  FIND LATETS SAMPLES FROM SINC BUFFER . */
    ADI_HAE_INPUTSAMPLE(DEV, (SINC_IMEAS_DATA[PWM_SINC_LOOP]), (SINC_VEXT_DATA[PWM_SINC_LOOP]));
     INDEX++;
      IF (INDEX >= NUM_SAMPLES) INDEX = 0;
   }
   COUNT++;
}
```

One Technology Way • P.O. Box 9106 • Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 • Fax: 781.461.3113 • www.analog.com

TA11824sc-0-9/13